Iterative Development

Project 2

Alex Harper

Contents

1 Introduction

Like the last paper, I am splitting the documentation I did into a different pdf to keep things neater.

This time was iteration 2, a continuation of the project with still a focus some on the use cases and getting the core sured up. Since my project is entierly about working with the remote servers for reddit, almost all the work is in extending the api headers.

2 History of Work

This probably will be the more entertaining part of my paper, so lets do it first. I had 3 major arcs of effort while doing my code up to the end of the work period I gave myself.

2.1 First Work

I started out with trying to implement what I had first came up with and drawn. I had this great idea of having a base class for all the reddit objects which enabled me to do other things that I had planned on. Luckily i didn't go crazy with the planning as it didn't work out.

The base class was going to handle pre-parsing all the JSON that the servers threw back at me. For all the data it would break up the key/value pairs, the sub-objects, and the arrays. But the nefarious issue was that arrays were not constrained to only hold values, but could hold any of the 3 things, and mix and match them at will. This made it too complicated to support and would have been harder to use *after* the parsing.

So out the window goes the idea of pre-parsing. I decided that it couldn't be that hard to simply hard code what I am looking for into the individual different objects. I later found out that I was right, and that directly embeding the domain knowledge into the parsing was the right thing to do.

2.2 Second Considerations

So with the central parsing gone, I had no real reason to have a common base for the objects anymore since they shared very little in common. With that, I deleted the whole thing. It felt good to just remove the not working code and be done with it even after spending a couple days fighting it. But then I was left with a problem, the central base was something I had planned a

few things on before. For example, a comment could have a parent of either another comment or a link. I was going to store the parent as a pointer to the base class, but then I got to thinking about what that would *mean*.

If the objects stored a pointer to their parent, that means the parent has to exist. How was the lifetime of this parent going to be managed, why does it need to be a full object, what would the child even need to knows who its parent is? Besides having orphaned children, there was no reason to have a full object instance that I could think of since each thing (term used on the website) had an id to reference it by. A child could simply remember the id of the parent, and then if we need an object later, we could use the id somehow to make it.

And there was the main consideration I have when trying to make generic tools as I go; what does it mean for the client program? If I was walking into this situation, and it somehow worked, would I be happy with the way things are given to me? I thought of the situation with the user inbox. Each thing in the inbox could be one of several types (message, comment, link), and I had intended to hand back a single list holding everything. If I was the client, would it not be tedious to have to check what each thing is, cast the objects, and then use them? After some thought, I decided it would be better to simply have 3 lists, each would hold each of the items; the order does not matter, and if they needed that, everything has a time code on it.

And so I started in on the code again, scrubbing the old concepts out. By the time I finished, well, the only thing to properly survive was the session manager which had been fully independent (as intended). That isn't saying too much since a lot of the effort was in getting the JSON pre-parsing to work...

2.3 Third time is the Charm

And then with not much vigor, it was time to try again! With only my trusty session manager to aid me, and unknown lands before me, the quest of deciphering what to do began. To start of simple, I wanted to see what I was given when asking data about a subreddit. What came back was "technically" human readable, but there was no spacing or newlines, and so a confusing mess.

The initial work of figuring out what to even do with the server response was slow because I would paste the output into a text editor, and by hand add spacing in. I did this a couple times, and looked through the things it had. I went into my code, fed it into the Qt JSON parsing stuff, and fooled around until values were being dumped into variables. While I didn't know what all the fields were (not even their types if they were null), I got most of

the information figured out.

Then I tried to work on parsing the posts on a subreddit. When looking at just the subreddit info, it was not too bad, and not a crazy number of items, but with the list of posts, it was untenable. That is when I finally got around to writing a recursive printing function that turned into two function because arrays are special little things. This made it a lot easier to rapidly test different places, and see what things changed, what stayed the same, and what I needed to do to parse it.

I also work on things after that, all fairly repetative looking through JSON and dumping values into variables.

2.4 Where I Am Now

So with the story of what i did out of the way, what does my program do now?

- Handles a user session
- Enforces basic API rules
- Reads JSON from the server
- Subreddit object parses JSON
- Subreddit object returns lists of posts
- Subreddit object can post a new link or text
- Post object parses JSON either from the subredit or directly from a url
- Post object returns lists of comments
- Post object can post a new comment
- Comment object parses JSON from post directly or from a url
- Comment object can post a reply for itself
- Account object can parse JSON about the current user

And what I am planning on trying to do next time (sprints are they called?)

- (mild) Get the imgur API working (simple in comparison to reddit)
- (easy) Read the user inbox (probably will be part of Account)

- (easy) Parse the messages in the inbox
- (mild) Bring code from my old project to mess with the images that are downloaded and then upload them to imgur

Just want to point out how silly it feels doing the JSON parsing. Below is a common line of code.

```
parseJson(top.array()[0].toObject()["data"].toObject()["children"]
    .toArray()[0].toObject()["data"].toObject());
```

3 Documentation

This is a bit short because I should be putting most of the relavent comments into the other document. But I would like to mention that the other doc has both old and new diagrams of things, just to show how things changed as I went on.

A code

Ok, so the code is 1694 LoC last time I checked (using some tool I found). That is a little silly to copy and paste here, and honestly, why is everyone doing that except to pad length? So just assume it is 30 pages of code, and you can just check my github page for the code itself.

B Reddit Post

I had made some posts and comments and such while debuging my bot. Since those were not pure tests, here is a thread where the bot had done 100% of the work. Here the bot made the thread, and then also put all those comment in. As you can see, it handles Utf8 and has no problem with the special markdown format.